	U		Document ID	Issue Date	Pages	Title	Current OR	Current XRef	Retrieval Classif	Inventor	S	P	2	3	4	5
14	☒	☒	US 4194242 A	19800318	6	Method and system for determining interest rates	705/38	235/379	705/38	Robbins, Robert						

different proto implementations having the same name. Each row of the Node Descriptions table **124** uniquely corresponds to one proto node. The Node Descriptions table **124** has columns named ID, Node Name, World ID, and Is Grouping. The ID column stores an identifier value that is uniquely associated with one proto node. The Node Name column stores the name of the proto node. The World ID stores an identifier of a world that is associated with the node, or in which the proto node is defined. The Is Grouping column stores a value, such as a Boolean value, that indicates whether the node is a "grouping" node that can have other nodes as children.

An IS Mapping table **126** is used to store IS mappings for proto implementations. Within a PROTO definition, according to the VRML language definition, the IS statement is used to associate a specified field with a field value that is defined in the PROTO header. The IS Mapping table **126** has columns named World ID, Node ID, Field Name, and Mapped Name. The World ID, Node ID, and Field Name values are used to uniquely identify a given node field as having an IS mapping. The Mapped Name value defines the value of the IS mapping.

MISCELLANEOUS TABLES

The database schema also includes tables for handling miscellaneous features and functions. For example, the VRML language defines a USE keyword that can be placed at any place in a VRML world definition in which a node definition is permitted. The USE keyword provides a way to refer to a previously named and defined node so that the referenced node does not have to be repeated in the VRML file. Accordingly, the schema provides a way to store USE references, in a Use table **125**. Each row of the Use table **125** is associated with one USE keyword. The Use table **125** has columns named World ID, Node ID, and Node Name. Values in the World ID identify the world with which the current row is associated. A USE node will have an entry stored in the Nodes table **106** with a value of "USE" in the Node Name column. In the Use table **125**, the values of the World ID column and the Node ID column match corresponding values in the Nodes table. The Node Name column stores the actual text value for the name of the node referenced by the USE keyword. Thus, in the schema, instances of USE keywords in a VRML file are treated like nodes. The Node Name column stores the name of the node referenced by the USE keyword.

The database schema provides a way to generate nodes dynamically from the database **20**. A Dynamic Children table **128** is used to track nodes that have children with fields that are dynamically generated using SQL queries applied to the external database **22**. Each row of the Dynamic Children table **128** represents a node having a child node that has at least one dynamically generated field. The Dynamic Children table **128** has columns named Parent World ID and Child World ID. The Parent World ID column stores a value referencing a parent world or parent node. The Child World ID column stores a value identifying the child node that has the dynamically generated field. The Dynamic Children table **128** is used by a graphical user interface generating utility to distinctly display all nodes containing dynamically generated values.

A Raw table **130** is used to store the raw text of the VRML source file on the server that is decomposed into the database **20**. Each row of the Raw table **130** corresponds to a block of text in the original VRML world or source text. The Raw table **130** has columns named ID, Text, Start Line In World, and Line Count. The ID column stores a unique identifier associated with a group of text. The Text column is a string field that stores text read from the VRML source file. The Start Line In World column stores a numeric value indicating the line number within the VRML source file at which the group of text stored in the Text column begins. The Line Count column stores a numeric value indicating the number of lines in the source file that are represented by the text that is stored in the Text column.

In an embodiment of the invention, zero or more portions of a VRML world are stored in the form of raw text in the Raw table **130**, and one or more portions of the world are stored in decomposed form in the other tables that have been described above. In this embodiment, when a VRML world is to be rendered, mechanisms of the embodiment deliver the portions of the world from the Raw table **130** and the other tables **108** to the browser **26**. This allows the implementation to optimize the delivery of the world by automatically combining the original text with decomposed portions. In this context, the term "component" is also used to refer to a portion of a world. A component is a group of VRML text, or one or more decomposed nodes stored in the above-described tables **108**.

To facilitate this process, the database **20** includes a Composition table **132** that describes the portions of the world and their characteristics. Each row of the Composition table **132** corresponds to a component of the world. The Composition table **132** has columns named ID, Component Type, Component ID, Start Line, End Line, Start Character, End Character, Parent ID, and Parent Field. The ID column stores an identifier value that is uniquely associated with a row of the Composition table **132**. Thus, the ID column serves as a row identifier.

The Component Type column stores a value indicating whether the component stored in the current row is a raw text chunk or a decomposed world. In an embodiment, the value 1 indicates a decomposed world, and the value 2 indicates a raw text chunk. The Component ID column stores an identifier that is uniquely associated with a component. In an embodiment, the Component ID value is unique to a component, whereas the ID value is unique to a row. Thus, a particular component may be represented in more than one row, each of which stores the same Component ID.

When the component is a decomposed world, the Component ID value matches or references a World ID or External ID value stored in the Description table **104**. Also in that case, the Parent ID and Parent Field columns provide information referencing the parent node of the component. Thus, to deliver a decomposed component to the VRML interpreter **28** of the browser **26**, the generating mechanism discussed further below reads values from the Parent ID and Parent Field columns, looks up the node identified by the Parent ID in the Nodes table, builds the required text, and delivers it to the browser **26**.

When the component is raw text that is stored in the Raw table **130**, the Component ID value of the Composition table matches or references an ID value in the Raw table. Also, in this case, the Start Line, End Line, Start Character, and End Character columns store numeric values that define a range of text to be delivered. In particular, these columns respectively store the starting line number, ending line number, starting character number within the starting line, and ending character number within the ending line for the text that is to be delivered. Thus, to deliver a raw text component to the browser **26**, the mechanisms discussed further below read the values of the Start Line, End Line, Start Character, and End Character columns, locate the text file that stores the raw text of the world on the server **10**, and read